Exploring the Parameter State Space of Stacking

Alexander K. Seewald ALEXSEE@OEFAI.AT; HTTP://ALEX.SEEWALD.AT
Austrian Research Institute for Artificial Intelligence, Schottengasse 3, A-1010 Wien, Austria

Abstract

Ensemble learning schemes are a new field in data mining. While current research concentrates mainly on improving the performance of single learning algorithms, an alternative is to combine learners with different biases. Stacking is the best-known such scheme which tries to combine learners' predictions or confidences via another learning algorithm. However, the adoption of Stacking into the data mining community is hampered by its large parameter space, consisting mainly of other learning algorithms: (1) the set of learning algorithms to combine, (2) the meta-learner responsible for the combining and (3) the type of meta-data to use: confidences or predictions. None of these parameters are obvious choices. Furthermore, little is known about the relation between parameter settings and performance of Stacking. By exploring all of Stacking's parameter settings and their interdependencies, we intend make Stacking a suitable choice for mainstream data mining applications.

1. Introduction

When faced with the decision "Which algorithm will be most accurate on my classification problem?", the predominant approach is to estimate the accuracy of the candidate algorithms on the problem and select the one that appears to be most accurate. Schaffer (1993) has investigated this approach in a small study with three learning algorithms on five UCI datasets. His conclusions are that on the one hand this procedure is on average better than working with a single learning algorithm, but, on the other hand, the cross-validation procedure often picks the wrong base algorithm on individual problems. This problem is expected to become more severe with an increasing number of classifiers.

As a cross-validation basically computes a prediction for each example in the training set, it was soon realized that this information could be used in more elaborate ways than simply counting the number of correct and incorrect predictions. One such ensemble learning method or meta-classification scheme is the family of *stacking* algorithms (Wolpert, 1992). The basic idea of Stacking is to use the predictions of the original classifiers as attributes in a new training set that keeps the original class labels. Stacking thus utilizes a *meta* classifier to combine the predictions from several *base* classifiers. Potentially, any classifier can be used as base and/or meta classifier. We shall refer to the type of meta-data consisting of base classifiers predictions as *preds*.

A straightforward extension of this approach is using class probability distributions of the original classifiers[1] which convey not only prediction information, but also confidence for all classes. We shall call the meta-data of this extension *class-probs*. This approach was evaluated and found to be superior to Stacking with predictions in (Ting & Witten, 1999), provided *multi-response linear regression* (MLR) is used as meta classifier.[2]

While approaches such as boosting and bagging, which combine classifiers of the same type, have been used extensively in data mining, Stacking has not. In terms of performance on our datasets based on significant differences[3], the most recent scheme StackingC[4] wins

[1] Every prediction is replaced by a vector of probabilities, consisting of one probability value for each class.

[2] In fact, what we noticed and will mention in more detail later is that they actually used meta-data similar to StackingC by Seewald (2002), but were unaware of its superiority – which can be seen by a note in their paper that both versions give comparable results. We will show that using classic *multi-response linear regression* yields a fairer comparison where *preds* and *classprobs* are competitive – no clear superiority either way is observed then.

[3] We used the χ^2 test after McNemar with 95% significance level, see Section 3

[4] i.e. Stacking with a fixed specialized meta-classifier based on MLR and the four diverse base classifiers mentioned in Section 3

five times and loses only once against AdaBoostM1 with C4.5 as base classifier; wins six times and never loses against Bagging implementation with C4.5-clone as base-classifier; wins three times against selection by crossvalidation (X-Val) and never loses; and wins five times against majority vote while losing just once. So StackingCseems to perform slightly better than its competitors, even though much less research has been focused on improving Stacking! Why then has Stacking not been adopted more widely? Tentatively we can suggest some reasons: For one, Stacking requires an integrated workbench including common machine learning classifiers. As of the time of writing this paper, two of the largest commercially available data mining tools lack a basic machine learning classifier, NaiveBayes. Also, Stacking requires a lot of parameters: which base-classifiers to choose, which meta-classifier to choose and also the type of meta-data – either predictions[5] *preds* or complete probablility distributions[6] *class-probs*. We felt it was time to investigate parameter setting for Stacking systematically to see how various parameters contribute to Stacking's performance, in order to see where sensible areas for further improvement may lie, but also to give useful proposals for *all* parameter settings. We investigated both the original Stacking introduced in (Wolpert, 1992) and the extension by Ting and Witten (1999) here. In some cases we also relate their performance to that of the most recent scheme, StackingC.

2. Overview

At first, Section 3 will present the experimental setup, the datasets and classifiers considered and details on significance tests and alpha-confidence levels.

In Section 4 we address the choice of base classifiers, using MLR as meta classifier as proposed by (Ting & Witten, 1999). We show that it is quite hard to significantly beat the trivial choice of using all available base classifiers and even harder to beat an informed choice of four base classifiers chosen via a priori and a posteriori arguments on diversity and base classifier performance. This makes base classifier choice the least influential factor on Stacking's performance. Intuitively, we would also expect that the *know-how* to combine the output of classifiers is more important than *which* classifiers to combine, as long as a reasonably large and diverse set is chosen.

[5] as in the original proposal by Wolpert (1992)

[6] as the extension proposed by Ting and Witten (1999).

In Section 5 we address both meta classifier choice and type of meta-data[7] to be used, on two subsets of base classifiers. We show that MLR is indeed the best classifier for *preds* meta-data, among those we considered. We notice that the performance differerences of those variants using *preds* meta-data are much smaller than of the variants for *class-probs* meta-data, which indicates that the learning problem for *preds* is easier for most classifiers. NaiveBayes seems a reasonable if somewhat arbitrary choice for *preds* meta-data. At last we conclude that Stacking with predictions meta-data is competitive to using probability distribution meta-data. We point out that Ting and Witten (1999) may have used a variation of MLR similar in spirit to StackingC in their experiments which yields a biased comparison and may explain why their conclusion as to the merits of the different meta-data types differs from ours.

In Section 6, Related Research, we give a short overview on relevant research. Afterwards we conclude this paper with an outlook on the future of Stacking in data mining. We will now proceed to shortly characterize our experimental setup.

3. Experimental Setup

For our empirical evaluation we chose twenty-six datasets from the UCI Machine Learning Repository (Blake & Merz, 1998), shown in Table 1. These datasets include fourteen multi-class and twelve two-class problems. Reported accuracy estimates are from a single ten-fold stratified cross-validation. Significant differences were evaluated by a χ^2-test after McNemar[8] with significance level of 95%, unless otherwise noted.

As base classifiers for Stacking we considered the following seven base learners, which were chosen to cover a variety of different biases. For figures, classifier numbers are used instead of their proper names.

1. J48: a Java port of C4.5 Release 8 (Quinlan, 1993)

2. KStar: the K* instance-based learner (Cleary & Trigg, 1995)

[7] In the original introduction of Stacking by Wolpert (1992), the predictions of the base classifiers were combined (*preds*); Ting and Witten (1999) extended this to use complete probability distributions which also convey confidence (*class-probs*).

[8] Dietterich (1998) proposes this test when the investigated algorithm can be run only once.

Table 1. The used datasets with number of classes and examples, discrete and continuous attributes, baseline accuracy (%) and entropy in bits per example (Kononenko & Bratko, 1991).

Dataset	cl	Inst	disc	cont	bL	E
audiology	24	226	69	0	25.22	3.51
autos	7	205	10	16	32.68	2.29
balance-scale	3	625	0	4	45.76	1.32
breast-cancer	2	286	10	0	70.28	0.88
breast-w	2	699	0	9	65.52	0.93
colic	2	368	16	7	63.04	0.95
credit-a	2	690	9	6	55.51	0.99
credit-g	2	1000	13	7	70.00	0.88
diabetes	2	768	0	8	65.10	0.93
glass	7	214	0	9	35.51	2.19
heart-c	5	303	7	6	54.46	1.01
heart-h	5	294	7	6	63.95	0.96
heart-statlog	2	270	0	13	55.56	0.99
hepatitis	2	155	13	6	79.35	0.74
ionosphere	2	351	0	34	64.10	0.94
iris	3	150	0	4	33.33	1.58
labor	2	57	8	8	64.91	0.94
lymph	4	148	15	3	54.73	1.24
primary-t.	22	339	17	0	24.78	3.68
segment	7	2310	0	19	14.29	2.81
sonar	2	208	0	60	53.37	1.00
soybean	19	683	35	0	13.47	3.84
vehicle	4	846	0	18	25.41	2.00
vote	2	435	16	0	61.38	0.96
vowel	11	990	3	10	9.09	3.46
zoo	7	101	16	2	40.59	2.41

3. MLR: a multi-class learner which tries to separate each class from all other classes by linear regression (*multi-response linear regression*)
4. NaiveBayes: the Naive Bayes classifier using kernel density estimation over multiple values for continuous attributes, instead of assuming a simple normal distribution.
5. DecisionTable: a decision table learner.
6. IB1: the IBk instance-based learner with $K = 1$ nearest neighbors, in order to offset KStar with a maximally local learner.
7. KernelDensity: a simple kernel density classifier.

All algorithms are implemented in WEKA Release 3.1.8. Each of them returns a class probability distribution, i.e. they do not predict a single class, but give probability estimates for each possible class. Parameters for learning schemes which have not been mentioned were left at their default values.

These algorithms can be clustered into four natural groups by their internal structure, where the first member tends to give better results than the others.

- J48, DecisionTable
- NaiveBayes
- MLR
- KStar, IB1, KernelDensity

Surprisingly – as we found out during our experiments – this can also be supported empirically. In particular, the statistical correlation of accuracies within each group is always greater than 0.95 while it is much smaller between classifiers oferent groups. Thus, the structure of correlations allows us to determine these groups empirically.[9] So we considered not only the trivial set of all seven base classifiers, but also the set of four base classifiers J48, NaiveBayes, MLR and KStar by choosing from each correlated subgroup the classifier which performed best by geometric mean of accuracy ratio.

We define a variant as a specific Stacking algorithm of which all parameters – type of meta-data, meta-classifier, and the set of base classifiers – are known. Meta-data *class-probs* is signified by the prefix *St*, *preds* is signified by the prefix *StP*. After this prefix, *7B* refers to the full set of base classifiers while *4B* refers to the set of four diverse base classifiers mentioned in Section 3. After this, a hyphen precedes the meta-classifier's name or an abbreviation. E.g. *St7B-MLR* refers to Stacking with the full set of seven base classifiers, MLR as meta-classifier and *class-probs* as type of meta-data while *StP7B-MLR* refers to the same variant with *preds* as meta-data. In Section 5, we will define stacking groups as those variants which just differ in the meta-classifier. These are named without reference to the meta-classifier.

4. Base Classifier Choice

In this section we investigate Stacking variants with MLR as meta classifier[10] and any non-empty subset of our seven base classifiers as base classifiers[11]. Because of the large number of comparisons, we used a significance level of 99% here to reduce our alpha-error. If we consider using all seven base classifiers (*St7B-MLR*) as gold standard, about one tenth of our variants are significantly better than *St7B-MLR* on any dataset; the

[9] A preliminary analysis of the κ-statistic – a measure of diversity due to (Dietterich, 2000) – is also compatible with this finding: the inner-group diversity between classifiers from the same group is generally smaller than the diversity between classifiers from different groups.

[10] in the extension proposed by (Ting & Witten, 1999) – using probability distributions as meta-level data

[11] 128 variants per dataset

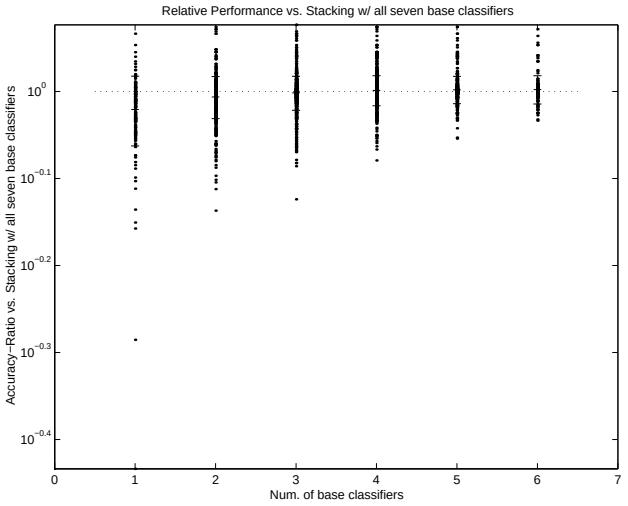

Figure 1. This figure shows the improvement of Stacking variants with different numbers of base classifiers and MLR as meta classifier, where all possible subsets of base classifiers were considered, as $\frac{Acc_{StVariant}}{Acc_{St7B-MLR}}$. The dotted line indicates a ratio of 1.0. Each • shows the ratio from one dataset and variant. Average and standard deviation over all subsets with the same number of base classifiers are shown as error bars.

rest shows no significant differences or are frequently even significantly worse. Figure 1 shows all our variants as accuracy ratios vs. *St7B-MLR*. One variant corresponds to a specific instantiation of Stacking with a non-empty subset from our seven base classifiers, always with MLR as meta-classifier, on *class-probs* meta-data. We have grouped the variants according to the size of the subset of base classifiers, from one to six.

Although at least some variants seem to offer considerable improvements on accuracy, this by no means assures a significant difference. We have considered two approaches to take significance into account.

At first we determined those variants which, over all datasets, never lose significantly against *St7B-MLR* and win as often as possible. It turns out that the maximum number of wins is only three, i.e. 11.5% of our datasets, which is somewhat disappointing. If we compare against *St4B-MLR*[12] instead, the maximum number of wins falls to zero. Concluding, dataset-independent variant resp. base classifier choice is not able to improve upon *St4B-MLR* here, even by hindsight.

[12] Stacking with the set of four diverse base classifiers we mentioned earlier. As we will see, this variant is slightly better than *St7B-MLR*.

If we were to consider a dataset-dependent choice of variants resp. base classifiers, i.e. possibly choosing a different variant for each dataset, the results are still quite disappointing. While the maximum number of wins is 7 (26.9%) vs. *St7B-MLR*, it is only 2 (7.7%) vs. *St4B-MLR*.[13] Thus, even if we would resort to meta-learning and found a model which lets us choose variants as good as by hindsight – which is doubtful to say the least – the improvements would still be quite insignificant.

So, in the remainder of this paper, we just consider two sets of base classifiers: the trivial choice of using all seven (those used by *St7B-MLR*), but also the subset of four diverse ones (those used by *St4B-MLR*) we mentioned in the last section. This is motivated by the wish to investigate both meta classifier choice and base classifier choice, even if quite coarse-grained, in one setup.

5. Meta Classifier Choice

In this section, we investigate Stacking variants along three independent dimensions – which are indeed *all* possible dimensions for Stacking's parameters.

1. Different Meta-Classifiers, chosen from our set of four diverse classifiers from Section 3 (numbers 1-4).

2. Type of meta-data: either predictions = *preds*[14] or complete class probability distributions = *class-probs*[15].

3. Base Classifiers: the set of four resp. seven base classifiers from Section 3, as maximally coarse-grained base classifier choice.

What concerns us most of all are the first two dimensions, i.e. to determine which meta-classifier is best, depending on type of meta-data; and whether or not one type of meta-data can be considered unconditionally superior. However, the third dimension can still roughly tell us the susceptibility of each meta-classifier to changes in its set of base classifiers and thus give us the opportunity to also investigate coarse-grained base-classifier choice at little additional cost.

The second and third dimension gives the name to our Stacking group – i.e. *St4B*, *St7B* for *class-probs*

[13] The maximum number of wins is even just 1 (3.9%) for StackingC with the full set of seven base classifiers.

[14] as in the original version of Stacking due to (Wolpert, 1992)

[15] as in the extension by (Ting & Witten, 1999)

meta-data[16] and *StP4B*, *StP7B* for *preds* meta-data. The meta-classifier is usually shown as an additional dimension in tables or figures, so all parameters are hereby accounted for.

To determine which meta-classifier is best, we resorted to a standard ranking of all meta-classifiers, separately for each stacking group. Table 2 shows *wins* minus *losses* of each meta-classifier versus all other meta-classifiers, for each group separately.

Table 2. This table shows the significant wins minus losses for each variant, against the other three meta-classifiers.

Variant	J48	KStar	MLR	NB
St7B	2	-26	16	8
St4B	-2	-19	19	2
StP7B	-1	0	1	0
StP4B	2	-9	1	6

A short glance at the table reveals some insights: For *class-probs* meta-data MLR is clearly the best meta-classifier, which was also shown by Ting and Witten (1999). However, for *preds* meta-data *StP7B* and *StP4B* disagree – the latter considers NaiveBayes superior while the former prefers MLR; with NaiveBayes and KStar both on a very close second place. The wins and losses are smaller than for *St7B* and *St4B* which indicates that meta-classifier choice seems to make less difference for predictions meta-data. This is to be expected since most classifiers are best suited to process nominal data, thus their performance as meta-classifiers is better and – since there is an upper limit on accuracy – tends to be more similar than for continuous data.

If we had to choose any one classifier for prediction meta-data, NaiveBayes seems the logical choice – it is not inconceivable that NaiveBayes ended up on second place because the ranking may be a little noisy; after all the difference is only one. A priori, the Bayesian approach inherent in NaiveBayes seems a suitable way to combine confidences from the base classifiers, but in practice this seems to work well only for *preds* meta-data. It may be that the modelling of continuous attributes by multiple kernels is not appropriate for this task and that a simple normal distribution may be more useful. This would explain why in our case, NaiveBayes performs clearly better with *preds* – as does J48 – and not equally well on *class-probs* and *preds*, as Ting and Witten (1999) found. Interestingly, both KStar and IB1 do not perform better on *class-probs* meta-data – rather, they are those meta-classifiers with the highest number of significant losses on *class-probs* vs. *preds*. This also contradicts Ting and Witten (1999), who concluded that IB1 performs better on *class-probs*. Further work is needed to resolve these contradictory results.

Now we will investigate whether one type of meta-data can be considered superior. For this, we determined accuracy ratios of *St7B* vs. *StP7B* and *St4B* vs. *StP4B* respectively, see Figure 2 and 3. The ranking in Table 3 shows it more clearly: While MLR performs better on *class-probs* meta-data, all other considered meta-classifiers perform better on *preds*, i.e. nominal, meta-data. The latter is to be expected since most machine learning algorithms are best suited to process nominal data.

This seems to contradict Ting and Witten (1999) who concluded that no meta-classifier on prediction meta-data offered satisfactory performance. However, their definition of MLR is different: each linear model had as input data only those partial class probability distributions concerned with the class it was trying to predict. So the dimensionality of the input data was smaller for MLR by a factor equal to the number of classes, which makes the mentioned comparison somewhat biased. If we try to replicate their results and replace MLR with their modified version, it is easily the best meta-classifier with a large margin – which explains their early dismissal of Stacking with *preds* meta-data. Concluding, Stacking with probability distributions (*class-probs*) is competitive to classic Stacking with predictions (*preds*). However, some Stacking variants based on *class-probs* perform significantly better than both, as shown in (Seewald, 2002). That is not to say that Stacking with predictions cannot be improved in the future – it remains an interesting research topic and should not yet be dismissed.

Now we were interested in the difference between using four base and all our seven base classifiers, see Figure 4 and 5. In the former case, we see that *St4B* is usually better and in four cases even significantly better when using MLR as meta classifier. In the latter case we see the same picture, but even less difference.[17] So we tenatively conclude that on average Stacking does seem to work better with a smaller set of less similar base classifiers, especially when using continuous meta-level data.

[16] which we consider to be the default case

[17] We noted that for StackingC, the difference is even less, i.e. 1.0012±0.0117 for the accuracy ratio of 4B vs. 7B – making it the scheme least susceptible to our coarse base classifier choice.

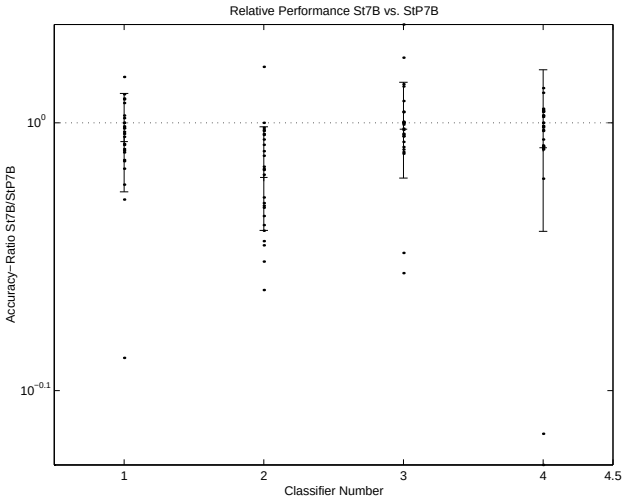

Figure 2. This figure shows the improvement of *St7B* over *StP7B*, i.e. using probability distributions meta-data vs. prediction meta-data, as $\frac{Acc_{St7B}}{Acc_{StP7B}}$. The dotted line indicates a ratio of 1.0. Each • shows the ratio from one dataset. Average and standard deviation over all datasets are shown as error bars.

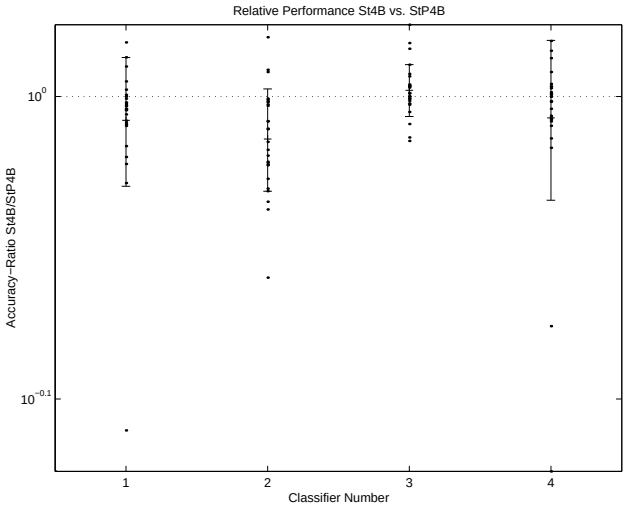

Figure 3. This figure shows the improvement of *St4B* over *StP4B*, i.e. using probability distributions meta-data vs. prediction meta-data, as $\frac{Acc_{St4B}}{Acc_{StP4B}}$. The dotted line indicates a ratio of 1.0. Each • shows the ratio from one dataset. Average and standard deviation over all datasets are shown as error bars.

Table 3. This table shows the improvements of Figures 2, 3, 4 and 5, in that order, in terms of significant wins and losses for the first mentioned variant.

Comparison	J48	KStar	MLR	NB
St7B vs. *StP7B*	1/5	1/14	4/3	1/4
St4B vs. *StP4B*	1/4	1/11	2/0	1/4
St4B vs. *St7B*	2/2	3/2	4/0	1/2
StP4B vs. *StP7B*	2/0	1/3	2/1	3/4

6. Related Research

Seewald (2002) investigates Stacking in the extension proposed by Ting and Witten (1999). He claims a weakness of this extension which is not apparent in the original version of Stacking and introduces a new variant, StackingC, in order to compensate for this weakness. Empirical evidence is given and supports these claims. An analysis into the reasons for improvement yields some interesting insights, most notably that the reason for this improvement is not mainly the dimensionality reduction of the meta-dataset, but also the higher diversity of the class models.

Dzeroski and Zenko (2002) investigate Stacking in the extension proposed by Ting and Witten (1999). They conclude that, when comparing against other meta-classification schemes, Stacking with MLR as meta-classifier is at best competitive to selection by crossvalidation (X-Val) and not significantly better as some papers claim, while their new variant sMM5 clearly beats X-Val. They propose a comparative study to resolve these contradictions in the literature.

Seewald and Fürnkranz (2001) propose a scheme called Grading that learns a meta-level classifier for each base classifier. Grading trains a meta classifier for each base classifier which tries to predict when its base classifier fails. This decision is based on the dataset's attributes. A weighted voting of the base classifiers' prediction gives the final class prediction. The voting weight is the confidence for a correct prediction of a base classifier, which is estimated by its associated meta classifier.

Cascading by Gama and Brazdil (2000) is a related variant to Stacking where the classifiers are applied in sequence and there is no dedicated level 1 classifier. Each base classifier, when applied to the data, adds his class probability distribution to the data and returns an augmented dataset, which is to be used by the next base classifier. Thus, the order in which the classifiers are executed becomes important. Cascading does not use an internal cross-validation like most other meta-classification schemes and is therefore claimed to be at least three times faster than Stacking. On the other hand in Stacking the classifier order is not important, thereby reducing the degrees of freedom and minimizing chances for overfitting. Furthermore, cascading increases the dimensionality of the meta-dataset with each step whereas Stacking's meta-dataset has a dimensionality which is independent of the dimensionality of

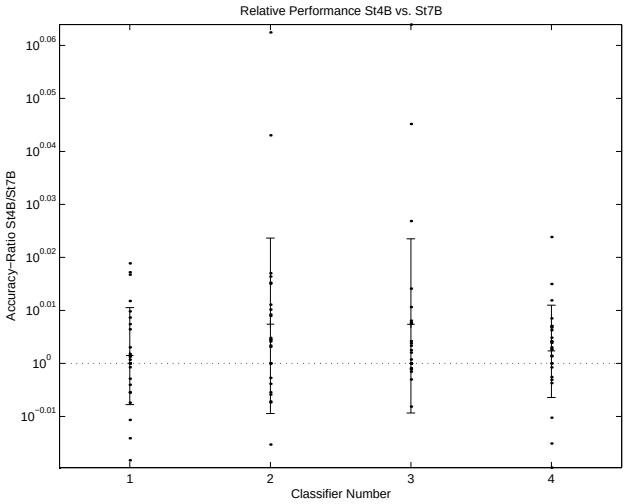

Figure 4. This figure shows the improvement of *St4B* over *St7B*, i.e. using only four base learners vs. using all seven for probability distributions meta-data, as $\frac{Acc_{St4B}}{Acc_{St7B}}$. The dotted line indicates a ratio of 1.0. Each • shows the ratio from one dataset. Average and standard deviation over all datasets are shown as error bars.

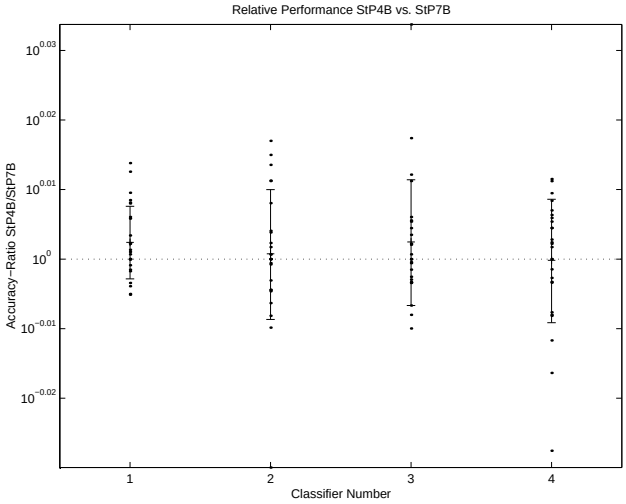

Figure 5. This figure shows the improvement of *StP4B* over *StP7B*, i.e. using only four base learners vs. using all seven for predictions meta-data, as $\frac{Acc_{StP4B}}{Acc_{StP7B}}$. The dotted line indicates a ratio of 1.0. Each • shows the ratio from one dataset. Average and standard deviation over all datasets are shown as error bars.

the dataset, i.e. the number of base classifiers multiplied with the number of classes.

Todorovski and Dzeroski (2000) introduce a novel method to combine multiple models. Instead of directly predicting the final class as all combining schemes we considered, their meta-learner MDT, based on C4.5, specifies which model to use for each example based on statistical and information theoretic measures computed from the class probability distribution. While their approach may make the combining scheme more comprehensible by learning an explicit decision tree decision tree, it is unclear whether this leads to better insight as well.

Merz (1999) studies the use of correspondence analysis and lazy learning to combine classifiers in a stacking-like setting. He compares his approach, SCANN, to two Stacking-variants with NaiveBayes resp. a backpropagation-trained neural network as meta-learner. MLR was not considered as meta-learner. According to experiments with synthetic data, his approach is equivalent to plurality vote if the models make uncorrelated errors. However, in practice this is seldom the case. Moreover, his approach is limited to using predictions as meta-level data and would fail for the class probability distributions which we use.

Ting and Witten (1999) deal with the type of generalizer suitable to derive the higher-level model and the kind of attributes it should use as input. They also investigated the usefulness of non-negativity constraints for feature weights within linear models, but found that it is not essential to get the best performance. However they found this may be useful to facilitate human comprehension of these models. Since our focus was on performance and not on comprehensibility, we did not use a non-negativity constraint.

Skalak (1997) includes an excellent overview about methods for constructing classifier ensembles. His other main contribution consists of investigating ensembles of coarse instance-based classifiers storing only a few prototypes per class.

Chan and Stolfo (1995) propose the use of *arbiters* and *combiners*. A combiner is more or less identical to stacking. Chan and Stolfo (1995) also investigate a related form, which they call an *attribute-combiner*. In this architecture, the original attributes are not replaced with the class predictions, but instead they are added to them. As Schaffer (1994) shows in his paper about bi-level stacking, this may result in worse performance.

7. Conclusion

We have explored the parameter state space of Stacking. Concerning the choice of base classifiers, we have found a set of four base classifiers, chosen by a priori and a posteriori arguments, which performs best. However, using all available base classifiers also remains an acceptable option, although the performance

may be worse[18] since the dimensionality of the meta-dataset is increased. When using predictions meta-data, the performance difference tends to be smaller, probably because most machine learning algorithms are better suited to deal with nominal data and therefore seem to exhibit less vulnerability to curse-of-dimensionality.

Concerning the choice of meta classifier and choice of meta-data to be used, we have found that MLR is indeed the best meta-classifier for probability distribution data. We showed probability distribution meta-data and predictions meta-data to perform comparably. For predictions meta-data, the best meta-classifier has less advantage because of less performance variation among the meta-classifiers. NaiveBayes is a reasonable choice since it is once on first and once on a very close second place.

However, given that both StackingC (Seewald, 2002) and sMM5 (Dzeroski & Zenko) outperform Stacking with *class-probs* meta-data and MLR as meta classifier, these two schemes can be considered the state-of-the-art in Stacking. Base-classifier choice is still applicable to both, so we propose using our set of four diverse base classifiers in both cases.

We believe that repeating our extensive base-classifier experiments with StackingC will not yield new insights. Because StackingC can be viewed as meta-classifier for probability distribution meta-data, we can see it as alternative meta-classifier. As such, we have investigated the distribution of accuracy ratios *St4B* by *St7B* and found it to be quite symmetric around 1.0, with the smallest standard deviation of all our meta-classifiers. Thus, StackingC seems to be least influenced by specific sets of base classifiers, which leads us to expect that base classifier choice has even less influence on StackingC than it has on Stacking. It still remains to be investigated whether this is also true for sMM5.

From preliminary experiments in (Seewald, 2002), StackingC and sMM5 should perform comparably, so performance-wise there is no reason to prefer the one or the other. However, because of the simpler meta classifier and lower-dimensional meta-dataset we would expect StackingC to be faster by at most an order of magnitude.

We hope that future research in Stacking will stay as exciting and interesting as it has been in the past and that these new variants, among with our proposals as to their parameters, will bring the best-known meta-classification scheme Stacking nearer to main-stream data mining.

[18] In our case, a penalty of at most four significant losses on twenty-six datasets is observed. Dependent on the type of meta-data and meta-classifier which is used, this may be much less – e.g. for StackingC, it is only one loss.

Acknowledgements

This research is supported by the Austrian *Fonds zur Förderung der Wissenschaftlichen Forschung (FWF)* under grant no. P12645-INF. The Austrian Research Institute for Artificial Intelligence is supported by the Austrian Federal Ministry of Education, Science and Culture. We would like to thank Saso Dzeroski and his colleagues for many interesting discussions; Elias Pampalk for valuable comments on a draft version of this paper and finally an anonymous reviewer of the Machine Learning Journal for valuable comments and suggestions.

References

Blake, C. L., Merz, C. J: UCI repository of machine learning databases. `http://www.ics.uci.edu/~mlearn/MLRepository.html` (1998). Department of Information and Computer Science, University of California at Irvine, Irvine CA.

Chan, P. K., Stolfo, S. J.: A comparative evaluation of voting and meta-learning on partitioned data. Proceedings of the 12th International Conference on Machine Learning (ICML-95) (pp. 90–98). Morgan Kaufmann, 1995.

Cleary, J. G., Trigg, L. E: K*: An instance-based learner using an entropic distance measure. In Prieditis, A., Russell, S., Proceedings of the 12th International Conference on Machine Learning (1995) 108–114, Lake Tahoe, CA.

Dietterich, T.G.: Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. Neural Computation, 10 (7) 1895-1924.

Dietterich, T. G: Ensemble methods in machine learning. In Kittler, J., Roli, F., First International Workshop on Multiple Classifier Systems (2000) 1–15. Springer-Verlag.

Dzeroski S., Zenko B.: Is Combining Classifiers Better than Selecting the Best One?, in Proceedings of the 19th International Conference on Machine Learning, ICML-2002, Morgan Kaufmann Publishers, San Francisco, 2002.

Gama J., Brazdil P.: Cascade Generalization, Machine Learning *41(3)*, 315-344, 2000.

Kononenko, I., Bratko, I: Information-based evaluation criterion for classifier's performance. Machine Learning *6* (1991) 67–80.

Merz, C.J.: Using Correspondence Analysis to Combine Classifiers, Machine Learning, *36(1-2)* (1999) 33–58.

Quinlan, J. R: C4.5: Programs for Machine Learning, Morgan Kaufmann, San Mateo, CA.

Schaffer, C.: Selecting a classification method by cross-validation. Machine Learning *13(1)*, 135–143, 1993.

Schaffer, C.: Cross-validation, stacking and bi-level stacking: Meta-methods for classification learning, in P. Cheeseman and R. W. Oldford (Eds.), *Selecting models from data: Artificial Intelligence and Statistics IV*, Springer-Verlag, 51–59, 1994.

Seewald A.K.: How to make Stacking Better and Faster While Also Taking Care of an Unknown Weakness, in Proceedings of the 19th International Conference on Machine Learning, ICML-2002, Morgan Kaufmann Publishers, San Francisco, 2002.

Seewald A.K., Fürnkranz J.: An Evaluation of Grading Classifiers, in Hoffmann F. et al. (eds.), Advances in Intelligent Data Analysis, 4th International Conference, IDA 2001, Proceedings, Springer, Berlin/Heidelberg/New York/Tokyo, pp.115-124, 2001.

Skalak, D. B: Prototype Selection for Composite Nearest Neighbor Classifiers. PhD Dissertation (1997), University of Massachusetts, Amherst, Massachusetts.

Ting, K. M., Witten, I. H: Issues in stacked generalization. Journal of Artificial Intelligence Research *10* (1999) 271–289.

Todorovski L., Dzeroski S.: Combining Multiple Models with Meta Decision Trees, in Zighed D.A. et al. (eds.), Principles of Data Mining and Knowledge Discovery, Lecture Notes in Artificial Intelligence, Springer-Verlag, pp.54-64, 2000.

Wolpert, D. H: Stacked generalization. Neural Networks *5(2)* (1992) 241–260.